

A Satisfiability Algorithm for Sparse Depth-2 Threshold Circuits

RUSSELL IMPAGLIAZZO * RAMAMOHAN PATURI *
STEFAN SCHNEIDER *

Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0404, USA
E-Mail: {russell, paturi, stschnei}@cs.ucsd.edu

November 2012

Abstract

We give a nontrivial algorithm for the satisfiability problem for cn -wire threshold circuits of depth 2 which is better than exhaustive search by a factor 2^{sn} where $s = 1/c^{O(c^2)}$. We believe that this is the first nontrivial satisfiability algorithm for cn -wire threshold circuits of depth 2. Our proof provides a characterization of the set of satisfying solutions of such a circuit as the union of the 0-1 solutions of at most $2^{(1-s)n}$ systems of linear equations. Our algorithm generalizes to arbitrary symmetric gates. It also applies to the special case of the feasibility of the 0-1 integer programming problem with linear size constraints. It is an independently interesting question whether there are nontrivial exponential time algorithms for integer programming. To our knowledge, our algorithm is the first to establish such an upper bound on the complexity of integer programming with linear size constraints.

One of our motivations is proving strong lower bounds for TC^0 circuits, exploiting the connection (established by Williams) between satisfiability algorithms and lower bounds. Our second motivation is to explore the connection between the expressive power of the circuits and the complexity of the corresponding circuit satisfiability problem.

A key idea underlying our algorithm is a novel random restriction technique where we use a game-theoretic argument to find a suitable parameter to simplify the circuit.

*This research is supported by NSF grant CCF-1213151 from the Division of Computing and Communication Foundations. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

1 Introduction

Satisfiability testing is both a canonical NP-complete problem ([1, 5]) and one of the most successful general approaches to solving real-world constraint satisfaction problems. In particular, optimized CNF-SAT heuristics are used to address a variety of combinatorial search problems successfully in practice, such as circuit and protocol design verification. The exact complexity of the satisfiability problem is also central to complexity theory, as demonstrated by Williams [8], who has showed that any improvement (by even a superpolynomial factor compared to exhaustive search) for the satisfiability problem for general circuits implies circuit lower bounds. Furthermore he has successfully used the connection to prove superpolynomial size bounds for ACC circuits using a novel nontrivial satisfiability algorithm for ACC circuits, solving a long standing open problem [9].

This raises the questions: For which circuit models do non-trivial satisfiability algorithms exist? How does the amount of improvement over exhaustive search relate to the expressive power of the model (and hence to lower bounds)? Can Satisfiability heuristics for stronger models than CNF's be useful for real-world instances?

Both the connection to circuit lower bounds and to heuristic search algorithms point to threshold circuits as the model to study next. Bounded-depth threshold circuits TC^0 are the next natural circuit class stronger than ACC_0 , and lower bounds are only known for the special case of depth 2 such circuits [2]. For larger depth, we have barely nonlinear lower bounds on the number of wires [4]. On the other hand, another very useful heuristic technique for search and optimization problems is Integer Linear Programming. Testing the feasibility of a region for a zero-one ILP is equivalent to testing satisfiability of a circuit with two levels, the bottom consisting of threshold gates and the top level being a conjunction. So both theoretical and real-world motivation points us to trying to understand the satisfiability problem for depth two threshold circuits.

In this paper, we give the first non-trivial Satisfiability algorithm for linear-sized depth two threshold functions, getting a constant savings in the exponent over exhaustive search. As a consequence, we also get a similar result for linear-sized ILP with small coefficients. Our algorithm also extends to arbitrary symmetric gates. We consider this just a first step to a real understanding of the satisfiability problem for threshold circuits.

2 Notation

Let V be a set of variables with $|V| = n$. An *assignment* on V is a function $V \rightarrow \{0, 1\}$ that assigns every variable a Boolean value. A *restriction* is an assignment on a set $U \subseteq V$.

A *symmetric gate* on n variables x_1, \dots, x_n is defined by *weights* $w_i \in \mathbb{Z}$ for $1 \leq i \leq n$ and an *output function* $g : \mathbb{Z} \rightarrow \{0, 1\}$. The output of the symmetric gate is $g(\sum_{i=1}^n w_i x_i)$. The *fan-in* of the symmetric gate is $\sum_{i=1}^n |w_i|$. In other words, we consider weights other than $-1, 0$, and 1 as multiple wires from the same variable. We call a variable an input to a gate if the corresponding weight is nonzero. For a given assignment b_1, \dots, b_n , we call the weighted sum $\sum_{i=1}^n b_i w_i$ the *value* of the gate. We also extend the definition of a symmetric gate to d -ary symmetric gates whose inputs and outputs are d -ary.

A *threshold gate* is a symmetric gate where the output function is a threshold function, i.e., $g(x) = 1$ if $x \geq A$ and $g(x) = 0$ otherwise for some threshold A .

For a collection of threshold gates, the *number of wires* is the sum of their fan-ins. A *depth-2 threshold circuit* consists of a collection of m threshold gates (called the *bottom-level gates*) on the same n variables and a threshold gate (called the *top-level gate*) on the outputs of the bottom-level gates. The output of the circuit is the output of the top-level gate. For a d -ary depth-2 threshold

circuit, the gates are d -ary gates and the top-level gate only outputs Boolean values. The number of wires of a depth-2 threshold circuit is the number of wires of the bottom-level gates. For all our circuits, we assume that the fan-in of the top-level gate is polynomially bounded although this condition can be relaxed considerably. We call a threshold circuit *sparse* if the the number of wires is linear in the number of variables.

A *satisfiability algorithm* for depth-2 threshold circuits is an algorithm that takes as input a depth-2 threshold circuit and outputs an assignment such that the circuit evaluates to 1 under the assignment.

A *linear function* on a variable set x_1, \dots, x_n is a function $g(x_1, \dots, x_n) = \sum_{i=1}^n w_i x_i$, where $w_i \in \mathbb{Z}$ are called the *coefficients*. The *size* of a linear function is the sum of the absolute values of the coefficients. A *linear equation* is an equation of the form $g(x_1, \dots, x_n) = a$ for some $a \in \mathbb{Z}$ and a linear g , and a *linear inequality* is an inequality of the form $g(x_1, \dots, x_n) \geq a$.

An algorithm for the *Integer Programming Feasibility Problem* takes as input a collection of linear inequalities on variables x_1, \dots, x_n , and outputs an assignment $\{x_1, \dots, x_n\} \rightarrow \mathbb{Z}$ such that all inequalities are satisfied. We call an inequality of the form $0 \leq x_i \leq d-1$ a *capacity constraint*. In a *0-1 Integer Programming Feasibility Problem* each variables is constrained to be 0 or 1.

We use $\tilde{O}(f(n))$ to denote the asymptotic growth of a function f ignoring polynomial factors. Informally, we say an algorithm is *nontrivial*, if its run time is significantly better than exhaustive search. If \mathcal{A} is a satisfiability algorithm for circuits with n variables with run time $\tilde{O}(2^{(1-s)n})$, we call s the *savings* of the algorithm over exhaustive search.

All logarithms are base 2 unless noted otherwise.

3 Results and Techniques

The main contribution of the paper is a nontrivial satisfiability algorithm for sparse threshold circuits of depth 2. More precisely, we prove the following:

Theorem 3.1. *There is a satisfiability algorithm for depth-2 threshold circuits on n variables with cn wires with running time $\tilde{O}(2^{(1-s)n})$ where*

$$s = \frac{1}{c^{O(c^2)}}$$

While the proof in Section 5 assumes a Boolean circuit for simplicity, the proof easily extends to threshold circuits with d -ary inputs, yielding the following corollary.

Corollary 3.2. *There is a satisfiability algorithm for depth-2 threshold circuits on n d -ary variables with cn wires with running time $\tilde{O}(d^{(1-s)n})$ where*

$$s = \frac{1}{c^{O(c^2)}}$$

In the following, we provide a high level description of our algorithm. The idea is to select a random subset U of variables of size $(1-p)n$ so that for each assignment to the variables in U , the circuit becomes *simpler*. In particular, for each assignment of U , we would like most of the bottom-level gates of the circuit to depend on at most one variable and that the number of *exceptional gates* (those that depend on more than one variable) to be small. For each exceptional gate (including the top-level gate), we guess its value, write a linear equation, and check whether a Boolean assignment to the variables simultaneously satisfies the linear equations using a nontrivial algorithm due to Williams [7], where he uses fast matrix multiplication algorithm to achieve constant savings in the

run time. However, we need the overhead for guessing the values to be smaller than the savings achieved with the fast matrix multiplication technique. For this, it is crucial that the number of exceptional gates is not too large.

One possible approach would be to select p so that there are few gates with fan-in larger than $\frac{1}{p}$. The intuition is that we can expect the gates with fan-in smaller than $\frac{1}{p}$ to depend on at most one variable after simplification. If the number of gates with fan-in larger than $\frac{1}{p}$ is not too large, we will succeed. However, it is not clear apriori that such a p exists. No matter what p one (*the algorithm designer*) selects, the *circuit designer* might select only gates with fan-in slightly larger than $\frac{1}{p}$, thus making the number of exceptional gates too large.

Our first successful attack is an analytical argument, which shows that for every threshold circuit with at most cn wires at the bottom-level, there is a $p > 0$ which is only doubly exponentially small such that there are not too many gates whose fan-in is slightly above $\frac{1}{p}$. Evidently, there cannot be too many gates with fan-in much larger than $\frac{1}{p}$, since the total number of wires is constrained. This argument yields a savings which is doubly exponentially small in c . However, it is useful to model the interplay between the parameter p and the circuit as an explicit zero-sum game, where the first player's (the algorithm designer) pure strategies are the values of p and the second player's (the circuit designer) pure strategies are the circuits where all the gates have the same fan-in. The payoff is the difference between the savings achieved by the Williams's algorithm and the overhead.

The mixed strategies of the circuit designer are threshold circuits with at most cn bottom-level wires, where each such circuit is viewed as a distribution of the total number of wires among gates of different fan-in. The mixed strategies of the algorithm designer are distributions on the values of p . We then apply the Min-Max theorem to lower bound the expected value of the game by exhibiting a distribution (with finite support) on the values of p . We then search through the values in the support of the distribution to find a p that produces the expected value. This game-theoretic analysis yields an overall satisfiability savings which is only single exponentially small in c .

We observe that the same ideas easily extend to arbitrary symmetric gates, as we only require that the value of the gate uniquely determine its output.

Since the Integer Programming Feasibility Problem with capacity constraints can be expressed as a depth-2 threshold circuit with an AND gate as the top-level gate, the results translate directly to the feasibility version of sparse Integer Programs with capacity constraints, and 0-1 Integer Programs in particular. We get

Corollary 3.3. *Let $\{g_1 \geq a_1, \dots, g_m \geq a_m\}$ be a collection of linear inequalities in variables $\{x_1, \dots, x_n\}$ with total size at most cn . There is an algorithm to find an integer solution to the linear inequalities with capacity constraints $0 \leq x_i \leq d - 1$ for all i with running time $\tilde{O}(d^{(1-s)n})$ for*

$$s = \frac{1}{c^{O(c^2)}}$$

The following three sections contain the details of the proof. Section 4 gives a nontrivial algorithm with constant savings for finding a Boolean solution to a system of linear equations. This algorithm follows the ideas of Williams [7], who provides a nontrivial algorithm with constant savings for the MAX-2-SAT problem using fast matrix multiplication technique. Section 5 presents the satisfiability algorithm for threshold circuits where the probability p is treated as a parameter. In section 6, we show how to select the parameter p using the Min-Max theorem to yield the claimed savings.

4 Solving Systems of Linear Equations

In this section we describe an algorithm to find a Boolean solution to a system of linear equations. The algorithm was presented by Williams [7] in the context of MAX-2-SAT, i.e., the problem of finding an assignment that satisfies the maximum number of clauses of a 2-CNF. We apply the algorithm to find Boolean solutions of quadratic equations.

A crucial component of the algorithm is fast matrix multiplication, and the runtime depends on the matrix multiplication exponent, the smallest ω such that two $n \times n$ matrices can be multiplied in time $O(n^\omega)$. The current best bound on ω is $\omega \leq 2.3727$ [10].

Lemma 4.1. *Let $f(x_1, \dots, x_n)$ be a degree two polynomial in n variables with integer coefficient between $-B$ and B . Assume f has no constant term. There is an algorithm which takes such an f and an $a \in \mathbb{Z}$ as inputs and decides whether $f(x_1, \dots, x_n) = a$ can be satisfied by a Boolean assignment in time $O\left(B^2 n^4 2^{\frac{\omega}{3}n}\right)$, where ω is the matrix multiplication exponent.*

Proof. Let $V = \{x_1, \dots, x_n\}$ be the set of n variables. For an assignment $\beta : W \rightarrow \{0, 1\}$, where $W \subseteq V$, let $\beta^* : V \rightarrow \{0, 1\}$ be the assignment that agrees with β on all variables in W , and sets all other variables to 0. Furthermore, let $f(\beta)$ be shorthand for $f(\beta^*)$. For two assignments $\beta : W \rightarrow \{0, 1\}$ and $\gamma : U \rightarrow \{0, 1\}$ on disjoint sets W and U , let $\beta\gamma : W \cup U \rightarrow \{0, 1\}$ be the assignment that agrees with β on W and with γ on U . If a monomial has both its variables in W , then it contributes to $f(\beta\gamma)$ as well as $f(\beta)$. On the other hand, if a monomial has one of its variables in W and the other in U , then it contributes to only $f(\beta\gamma)$.

Let V_1, V_2 and V_3 be a partition of V into sets of size $\frac{n}{3}$ each and let β be any assignment on V_1 , γ be an assignment on V_2 and δ be an assignment on V_3 . Then

$$\begin{aligned} f(\beta\gamma\delta) &= f(\beta\gamma) + f(\gamma\delta) + f(\delta\beta) - f(\beta) - f(\gamma) - f(\delta) \\ &= \left(f(\beta\gamma) - \frac{1}{2}f(\beta) - \frac{1}{2}f(\gamma)\right) + \left(f(\gamma\delta) - \frac{1}{2}f(\gamma) - \frac{1}{2}f(\delta)\right) \\ &\quad + \left(f(\delta\beta) - \frac{1}{2}f(\delta) - \frac{1}{2}f(\beta)\right) \end{aligned}$$

since each monomial makes a net contribution of one to each side of the equation. More specifically, every monomial contributes exactly once to the left side of the equation. If a monomial has both its variables in the same set, say V_1 , it contributes to $f(\beta)$, $f(\beta\gamma)$ and $f(\delta\beta)$ so that the algebraic contribution is 1. If a monomial has one in variable in one part and the other in a different part, say V_1 and V_2 , then it contributes exactly once to $f(\beta\gamma)$.

The algorithm computes three $2^{\frac{n}{3}} \times 2^{\frac{n}{3}}$ matrices B, C and D , where $B_{\beta,\gamma} = f(\beta\gamma) - \frac{1}{2}f(\beta) - \frac{1}{2}f(\gamma)$, $C_{\gamma,\delta} = f(\gamma\delta) - \frac{1}{2}f(\gamma) - \frac{1}{2}f(\delta)$ and $D_{\delta,\beta} = f(\delta\beta) - \frac{1}{2}f(\delta) - \frac{1}{2}f(\beta)$. We can construct these matrices in time $O(n^2 2^{\frac{2n}{3}})$ and the entries of the matrix are integers between $-Bn^2$ and Bn^2 . Furthermore, for a matrix M , let $M^{(m)}$ be the 0-1 matrix which is 1 at an entry if and only if the corresponding entry in M is m . There is an assignment $\alpha : V \rightarrow \{0, 1\}$ such that $f(\alpha) = a$ if and only if there are numbers b, c and d with $b + c + d = a$ and assignments β, γ and δ such that $B_{\beta,\gamma} = b$, $C_{\gamma,\delta} = c$, $D_{\delta,\beta} = d$. Therefore, for fixed numbers b, c and d , there are assignments β, γ and δ satisfying those conditions if and only if $(B^{(b)}C^{(c)})_{\beta,\delta} \geq 1$ and $D_{\delta,\beta}^{(d)} = 1$. Using a fast matrix multiplication algorithm with exponent $\omega = 2.3727$, we can multiply $B^{(b)}$ and $C^{(c)}$ in time $O\left(2^{\frac{\omega}{3}n}\right)$.

Since there are at most $4B^2n^4$ combinations of numbers b, c, d with $b + c + d = a$, the overall time complexity of the algorithm is $O\left(B^2n^4 2^{\frac{\omega}{3}n}\right)$. \square

Corollary 4.2. *Let $\mathcal{F} = \{f_1, \dots, f_m\}$ be a collection of linear functions on n variables with integer coefficients between $-D$ and D . Then we can find a Boolean solution to the system of equations $f_1(x_1, \dots, x_n) = 0, \dots, f_m(x_1, \dots, x_n) = 0$ in time $O\left(m^2 D^4 n^4 2^{\frac{\omega}{3}n}\right)$.*

Proof. The system of equations is equivalent to $\sum_{i=1}^m f_i^2(x_1, \dots, x_n) = 0$. For f_i^2 , the coefficients are between $-2D^2$ and $2D^2$. Since we sum over m quadratic functions, every coefficient in the quadratic equation is bounded by $2mD^2$ and the corollary follows from Lemma 4.1. \square

We use $s^\omega = 1 - \frac{\omega}{3}$ to denote the savings of the Williams's algorithm. We observe that the time complexity of the algorithm is $\tilde{O}\left(2^{(1-s^\omega)n}\right)$ for D and m polynomial in n .

5 The Algorithm

We develop the algorithm in three stages. In the first stage, we present a satisfiability algorithm for depth-2 threshold circuits with cn wires where all the bottom-level gates have the same fan-in f . The algorithm further depends on a parameter p and achieves savings $s_{p,f}$. For certain combinations of p and f the savings might be negative. In the second stage we extend the algorithm to threshold circuits with varying fan-in and show that the savings of the algorithm is a convex combination of $s_{p,f}$. In the final stage, in Section 6 we show how to select a p such that the savings is at least $\frac{1}{c^{O(c^2)}}$ for any distribution on f .

As we are mainly interested in the savings, we look at the logarithm of the time complexity and bound its expectation.

Lemma 5.1. *Let $0 \leq p \leq 1$ be a parameter and C be a depth-2 threshold circuit on variables $V = \{x_1, \dots, x_n\}$ with at most cn wires, and fan-in f for all bottom-level gates. There is an algorithm that decides the satisfiability for such C with time complexity T such that $\mathbf{E}[\log(T)] = (1 - s_{p,f})n$ for*

$$s_{p,f} = \begin{cases} \frac{s^\omega p}{2} & \text{if } pf < \frac{s^\omega}{2c} \\ s^\omega p - \frac{c}{f} \log\left(\frac{4}{s^\omega} c p f\right) & \text{otherwise} \end{cases}$$

Proof. We select a random subset $U \subseteq V$ such that a variable is in U with probability $(1 - p)$ independently. We note that $\mathbf{E}[|U|] = (1 - p)n$. For each of the $2^{|U|}$ assignments to U , we solve the satisfiability problem of the simplified circuit. Bottom-level gates where all inputs are in U are removed and the threshold of the top-level gate is adjusted appropriately. Gates that only depend on one input are replaced by a direct wire to the top-level gate with an appropriate weight and adjustment to the threshold of the top-level gate. For all gates with at least two remaining inputs, we guess the value of the gate and express the gate as a linear equation. Similarly, we guess the value of the top-level gate to get another linear equation. Using corollary 4.2 we solve the resulting system of linear equations on $n' = n - |U|$ variables in time $\tilde{O}\left(2^{(1-s^\omega)n'}\right)$ with savings $s^\omega = 1 - \frac{\omega}{3}$.

The critical part of the analysis is bounding the overhead from guessing the values of the gates. We first bound the number of distinct values a gate can take. The top-level gate can only take polynomially many different values. Consider a bottom-level gate with fan-in $l \geq 2$ after applying an assignment to the variables in U . We bound the number of distinct values that the gate can take in two different ways. The number of possible inputs, and hence the number of possible values is bounded by 2^l . On the other hand, since the value is an integer between $-l$ and l , the number of possible values for the gates is also upper bounded by $2l + 1$. Hence, we use $\min\{2^l, 2l + 1\}$ as an upper bound for the number of values of a bottom-level gate with fan-in l .

Since we have a control on the number of distinct values taken by a gate by assumption, our overhead crucially depends on the number of exceptional gates, gates that depend on more than one variable after applying an assignment to the variables in U . Intuition says that the number of exceptional gates should be small. If the fan-in of a gate is small, then we expect that it will be simplified to depend on at most one variable after assigning values to the variables in U . On the other hand, there cannot be too many gates of large fan-in. While the intuition is simple, it is tricky to make it work for us in the general context. At this stage, our focus is on estimating the savings $s_{p,f}$ for the probability parameter p and the fan-in f .

Let H be a random variable denoting the overhead. Our estimation of overhead and $s_{p,f}$ involves two cases. Let $t = \frac{s^\omega}{2c}$. We first consider the case $pf < t$. Let $U' \subseteq V - U$ be the set of variables that appear in exceptional gates. Our goal is to upper bound $\mathbf{E}[\log(H)] \leq \mathbf{E}[|U'|]$.

Consider a bottom-level gate. Let X be the random variable denoting the number of its inputs not in U . Let $f' \leq f$ be the number of variables the gate depends on, and let X be the random variable denoting the number of its inputs not in U . The distribution of X is $\text{Bin}(f', p)$, hence we have $\mathbf{E}[X] = f'p$. Let the random variable Y denote the number of variables that the gate can contribute to U' . Since U' is the set of variables appearing in exceptional gates, we have $Y = X$ for $X \geq 2$ and $Y = 0$ otherwise. Hence

$$\begin{aligned} \mathbf{E}[Y] &= \mathbf{E}[X] - \mathbf{P}[X = 1] \leq f'p - f'p(1-p)^{(f'-1)} \\ &\leq f'p(1 - (1-p)^{f'}) \leq f'p(1 - (1-f'p)) \\ &= (f'p)^2 \leq (fp)^2 \end{aligned}$$

by Bernoulli's inequality. Hence, for any variable x which is an input to the gate, the probability x belongs to U' is at most $\frac{\mathbf{E}[Y]}{f} \leq p^2 f \leq \frac{s^\omega p}{2c}$. Since the total number of wires is bounded by cn , we have $\mathbf{E}[\log(H)] \leq \mathbf{E}[|U'|] = cn \frac{s^\omega p}{2c} = \frac{s^\omega p}{2} n$.

For the logarithm of the time complexity this yields

$$\mathbf{E}[\log(T)] = \mathbf{E}[|U|] + \mathbf{E}[(1 - s^\omega)(n - |U|)] + \mathbf{E}\left[\frac{s^\omega p}{2} n\right] \leq n \left(1 - \frac{s^\omega p}{2}\right)$$

ignoring logarithmic factors from both the polynomial factors of Williams' algorithm and the overhead from guessing the value of the top-level gate. We have $s_{p,f} = \frac{s^\omega p}{2}$.

We now consider the case $pf \geq t$. Suppose the i -th gate has l_i inputs that are not in U . The expected value of l_i is pf . There are at most $2l_i + 1$ possible values for the gate. Since all the bottom-level gates have the same fan-in f , the number of bottom-level gates is at most cn/f and $\mathbf{E}[\sum_{i=1}^{cn/f} l_i] = pcn$. We bound the expected logarithm of the number of possible values of all gates by

$$\begin{aligned} \mathbf{E}\left[\log\left(\prod_{i=1}^{cn/f} (2l_i + 1)\right)\right] &= (cn/f) \sum_{i=1}^{cn/f} \mathbf{E}[(\log(2l_i + 1)f/cn)] \leq (cn/f) \log(2pf + 1) \\ &\leq (cn/f) \log\left(\frac{4}{s^\omega} cpf\right) \end{aligned}$$

where we use the concavity of the logarithm function in the penultimate step and the fact $pf \geq \frac{s^\omega}{2c}$ in the last step.

For the logarithm of the time complexity we get,

$$\mathbf{E}[|U|] + \mathbf{E}[(1 - s^\omega)(n - |U|)] + \mathbf{E}\left[cn/f \log\left(\frac{4}{s^\omega} cpf\right)\right] \leq n \left(1 - \left(s^\omega p - \frac{c}{f} \log\left(\frac{4}{s^\omega} cpf\right)\right)\right)$$

with savings $s_{p,f} = s^\omega p - \frac{c}{f} \log\left(\frac{4}{s^\omega} c p f\right)$. □

We now extend the algorithm to circuits with varying fan-in and show that the logarithm of the time complexities is lower bounded by a convex combination of the savings $s_{p,f}$. We model the cn -wire circuits of varying fan-in by a distribution \mathcal{F} on wires. For each fan-in f , the wire distribution \mathcal{F} specifies the number $c_f n$ of wires of bottom-level gates of fan-in f . We denote the savings of our algorithm on circuits with wire distribution \mathcal{F} by $s_{p,\mathcal{F}}$.

Lemma 5.2. *Let $0 \leq p \leq 1$ be a parameter and let C be a depth-2 threshold circuit on n variables with at most cn wires, where the wires are distributed according to \mathcal{F} . There is a satisfiability algorithm for such C with time complexity T such that $\mathbf{E}[\log(T)] = (1 - s_{p,\mathcal{F}})n$ for*

$$s_{p,\mathcal{F}} \geq \sum_{f=1}^n \frac{c_f}{c} s_{p,f}$$

Proof. The algorithm is the same as above. The logarithm of the overhead of guessing the values for all bottom-level gates with fan-in f is $\log(H_f) = \frac{c_f n}{f} \log\left(\frac{16}{s^\omega} c p f\right)$ if $p f \geq t$ and $\log(H_f) = \frac{c_f}{c} \frac{s^\omega p}{4} n$ otherwise. Solving the system of linear equations using corollary 4.2 and using linearity of expectation then yields the savings as claimed. □

6 The Algorithm as a Zero-Sum Game

The time complexity of the algorithm in Section 5 depends crucially on choosing a suitable parameter p . Instead of trying to directly determine a good parameter p by analyzing the wire distribution of the circuit, we apply a trick from game theory.

A zero-sum game with two players A and C is a game where both players pick a strategy and the outcome is determined by a function of the two strategies. Player A tries to maximize the outcome, while player C tries to minimize it. The Min-Max Theorem states that it does not matter which player moves first, as long as we allow mixed strategies for the players.

We model the task of choosing the parameter p as the following zero-sum game: Player A, the algorithm designer, picks some probability p , and player C, the circuit designer, picks a value f . The outcome is $s_{p,f}$, the savings of the algorithm. The algorithm designer tries to maximize the savings, and the circuit designer tries to minimize it. The wire distribution of a circuit is a mixed strategy for the circuit designer. A mixed strategy for the algorithm designer A would be a distribution on the probabilities.

A direct approach for designing the algorithm would be to select the parameter p depending on the circuit so that we obtain large savings. Specifically, given the wire distribution of the circuit \mathcal{F} , the algorithm designer picks a p and the outcome $s_{p,\mathcal{F}}$ is a convex combination of the values $s_{p,f}$. Using the Min-Max Theorem we turn this game around: The algorithm designer picks a mixed strategy and the circuit designer responds with a pure strategy f , a circuit where all bottom-level gates have fan-in f . The following lemma shows that there is a good strategy for the algorithm designer.

Lemma 6.1. *There is a distribution \mathcal{D} on parameters p such that for all f ,*

$$\mathbf{E}_{p \sim \mathcal{D}}[s_{p,f}] \geq \frac{1}{c^{O(c^2)}}$$

Proof. Let \mathcal{D} be the following distribution on p : For $I = O(c^2 \log(c))$ with suitable constants, and $1 \leq i \leq I$, we set $p = 2^{-i}$ with probability $A \cdot 2^{-(I-i+1)}$, where $A = \frac{1}{\sum_{i=1}^I 2^{-(I-i+1)}}$ is the normalization factor. We know that $1 \leq A \leq 2$. The expectation of p is $\mathbf{E}[p] = AI2^{-I-1}$.

Let f be any pure strategy of the circuit designer and $J = \log(f)$. The expected outcome of the game for these strategies is

$$\mathbf{E}_{p \sim \mathcal{D}}[s_{p,f}] = \sum_{i=1}^I 2^{-(I-i+1)} s_{2^{-i}, 2^J}.$$

To lower bound the expected outcome, we use a case analysis on the savings similar to the one in Section 5. Let $t = \frac{s^\omega}{2c}$ as defined in the previous section. Let $I' \leq I$ be the largest value such that for $i \leq I'$, we have $2^{J-i} \geq t$ and for $I' < i \leq I$ we have $2^{J-i} < t$.

Using the savings from lemma 5.1, we have $s_{2^{-i}, 2^J} = 2^{-i} s^\omega - \frac{c}{2^J} \log\left(\frac{4}{s^\omega} c 2^{J-i}\right)$ for $2^{J-i} \geq t$ and $s_{2^{-i}, 2^J} = \frac{2^{-i} s^\omega}{2}$ otherwise. The expected savings is then

$$\begin{aligned} \mathbf{E}_{p \sim \mathcal{D}}[s_{p,f}] &= \sum_{i=1}^I 2^{-(I-i+1)} s_{2^{-i}, 2^J} \\ &= \sum_{i=1}^{I'} 2^{-(I-i+1)} \left(2^{-i} s^\omega - \frac{c}{2^J} \log\left(\frac{4}{s^\omega} c 2^{J-i}\right) \right) + \sum_{i=I'+1}^I 2^{-(I-i+1)} \frac{2^{-i} s^\omega}{2} \\ &\geq \sum_{i=1}^I 2^{-(I-i+1)} \frac{2^{-i} s^\omega}{2} - \sum_{i=1}^{I'} 2^{-(I-i+1)} \frac{c}{2^J} \log\left(\frac{4}{s^\omega} c 2^{J-i}\right) \\ &= \frac{1}{2^{I+1}} \left(I \frac{s^\omega}{2} - c \sum_{i=1}^{I'} 2^{-(J-i)} \log\left(\frac{4}{s^\omega} c 2^{J-i}\right) \right) \end{aligned}$$

Let $j = \lceil (J-i) \rceil$. By the definition of I' we have $j \geq \log(t) = -\log(c) + \log\left(\frac{s^\omega}{8}\right)$. Hence

$$\begin{aligned} \sum_{i=1}^{I'} 2^{-(J-i)} \log\left(\frac{4}{s^\omega} c 2^{J-i}\right) &\leq \sum_{j=\log(t)}^{\infty} 2^{-j} \left(j + \log\left(\frac{4}{s^\omega} c\right) \right) \\ &\leq \frac{4c}{s^\omega} \log\left(\frac{4}{s^\omega} c\right) + \sum_{j=1}^{\infty} j 2^{-j} + \log\left(\frac{4}{s^\omega} c\right) \\ &= O(c \log(c)) \end{aligned}$$

Hence for $I = O(c^2 \log(c))$ we get

$$\mathbf{E}_{p \sim \mathcal{D}} s_{p,f} = \frac{1}{c^{O(c^2)}}$$

□

We now conclude that for every f there is a $p = 2^{-i}$ for $1 \leq i \leq I$, such that $s_{p,f} \geq \frac{1}{c^{O(c^2)}}$. Using that for every mixed strategy for f , the savings is a convex combination of the savings for pure strategies, we conclude the same for any strategy on f .

This gives us the final algorithm: Given a circuit C with wire distribution \mathcal{F} , evaluate $\mathbf{E}_{f \sim \mathcal{F}}[s_{p,f}]$ with $p = 2^{-i}$ for each $1 \leq i \leq I$ as above and use the optimal p for the random restriction.

The savings is tight in the sense that there is a mixed strategy on f such that the expected savings is at most $1/2^{\Omega(c)}$.

Lemma 6.2. *There is a wire distribution \mathcal{F} such that for any p*

$$\mathbf{E}_{f \sim \mathcal{F}}[s_{p,f}] \leq \frac{1}{2^{\Omega(c)}}$$

Proof. Let p be the strategy of the algorithm designer and let \mathcal{F} be the distribution such that for $1 \leq j \leq c$, $c_{2^j} = 1$ and $c_f = 0$ for any other f . By lemma 5.2 we have

$$\mathbf{E}_{f \sim \mathcal{F}}[s_{p,f}] = \sum_{j=1}^c \frac{1}{c} s_{p,2^j}$$

We argue that for large c and $p \geq \frac{1}{2^c}$, the savings is negative. Assume $p \geq \frac{1}{2^c}$. There is some $j^* \leq c$ such that for $f = 2^{j^*}$, $1 \leq pf \leq 2$. Using that for any p and f , the savings $s_{p,f}$ is upper bounded by $s^\omega p$ we get

$$\begin{aligned} \mathbf{E}_{f \sim \mathcal{F}}[s_{p,f}] &= \sum_{j=1}^c \frac{1}{c} s_{p,2^j} \\ &\leq s^\omega p - \frac{1}{c} s_{p,2^{j^*}} \\ &= s^\omega p - \frac{1}{c} \frac{c}{2^{j^*}} \log \left(\frac{4}{s^\omega} c p 2^{j^*} \right) \\ &\leq p \left(s^\omega - \frac{1}{2} \left(\log \left(\frac{4}{s^\omega} c \right) + 1 \right) \right) \end{aligned}$$

For large c , the expectation is therefore negative. On the other hand, if $p \leq \frac{1}{2^c}$, then $\mathbf{E}_{f \sim \mathcal{F}}[s_{p,f}] \leq \frac{s^\omega}{2^c} = \frac{1}{2^{\Omega(c)}}$. \square

7 Conclusions

In this paper, we present the first nontrivial algorithm for deciding the satisfiability of cn -wire threshold circuits of depth 2. The result extends to the more general case of circuits with symmetric gates. The same result also applies to the special case of 0-1 integer programming feasibility problem with sparse constraints. The algorithm improves over exhaustive search by a factor 2^{sn} where $s = 1/c^{O(c^2)}$.

Several straightforward open questions remain. Can we improve the savings for sparse depth-2 threshold circuits? The savings in our algorithm is exponentially small in c , while the best known savings for cn -size AC^0 circuits is only polylogarithmically small in c [3]. Can we decrease this gap? If not, can we explain it in terms of expressive power of the circuits?

Our algorithm handles only restrictive versions of threshold circuits of depth-2. Can we obtain nontrivial satisfiability algorithms for slightly more relaxed models? For example, it would be interesting to extend the result to depth-2 circuits with more wires. It would be very interesting to extend the result to larger depth circuits.

Our algorithm follows the general framework described by [6]: convert the input circuit to a simpler representation of the same function, and then solve Satisfiability for that representation. In [6], the representation was a decision tree. [3] used a more general representation, a partition of the Boolean cube into sub-cubes where the function was constant. Here, we are implicitly using an even more general representation: a partition of the cube into affine sub-spaces over the reals on which the function is constant. Unlike earlier work, the satisfiability problem is not easy even

for the simpler representation; instead of being polynomial time for each sub-space, we get only an improved exponential time. This motivates two further questions: What is the exact complexity of finding a Boolean point in a sub-space given as a system of linear equations? We reduce this problem to that of finding a solution to a 2-CSP satisfying exactly a given number of constraints, but there may be more direct approaches. Another question raised is to look at the size of such a partition as a measure of complexity for Boolean functions. Can we find explicit functions requiring almost exponentially many such sub-spaces for such a partition? Can we use this to prove lower bounds for threshold circuits?

Finally, our algorithm takes exponential space, because this is required for Williams' algorithm. Can we reduce the space requirement to polynomial space?

References

- [1] S.A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.
- [2] Andras Hajnal, Wolfgang Maass, Pavel Pudlak, Mario Szegedy, and Gyorgy Turan. Threshold circuits of bounded depth. In *Proceedings of the 28th Annual Symposium on Foundations of Computer Science, SFCS '87*, pages 99–110, Washington, DC, USA, 1987. IEEE Computer Society.
- [3] Russell Impagliazzo, Williams Matthews, and Ramamohan Paturi. A Satisfiability Algorithm for AC^0 . In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, 2012.
- [4] Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. Size–depth tradeoffs for threshold circuits. *SIAM J. Comput.*, 26(3):693–707, 1997. preliminary version published in STOC 1993.
- [5] L. Levin. Universal sorting problems. *Problems of Information Transmission*, 9:265–266, 1973.
- [6] Rahul Santhanam. Fighting perebor: New and improved algorithms for formula and qbf satisfiability. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science, FOCS '10*, pages 183–192, Washington, DC, USA, 2010. IEEE Computer Society.
- [7] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348:357–365, 2005.
- [8] Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC '10, pages 231–240, New York, NY, USA, 2010. ACM.
- [9] Ryan Williams. Non-Uniform ACC Circuit Lower Bounds. In *Proceedings of the Twenty-Sixth Annual IEEE Conference on Computational Complexity*, 2011.
- [10] Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In *Proceedings of the 44th symposium on Theory of Computing*, STOC '12, pages 887–898, New York, NY, USA, 2012. ACM.